Arrays of Objects

Parallel array solution:

```
rollNum=          {6016, 6024, 6078};
midsemMarks =   {61.7,  54 , 74.2 };
name =       {"Abhishek Murthy", "Abhishek Singh",
                    "Ankit"};
```

In practice, parallel arrays are hard to maintain.
Better: define class STUDENT with attributes

- rollNum,
- marks,
- name,

and define arrays of the STUDENT object: Student[ ]

Also, may need to return more than one parameter

- Want to identify both maximum marks, and also who has this maximum marks
  - Need to return two values from FindMax
  - Or return index and find each record

- Better: use Student as object

Student Class: Attributes

```
public class Student
{
    private String rollNum;
    private String firstName;
    private String midName;
    private String lastName;
    private int hostelNum;
    private char wing;
    private int roomNum;
    private Date birthDate;
    private Course[] courseList;

    private char grade;
    private double marks;
. . .
}
```

see file: Student.java

Other Classes used by “Student”

Note:

```
        private Date birthDate;
```

Requires class “Date”. This is defined in separate file **Date.java** in the same directory.

Similarly

```
    private Course[] courseList;
```

Requires the class “Course”, defined in **Course.java**.

These three files constitute a “package”.

If they are in the same directory, Java automatically recognizes them as a package

Please save these files to the same directory and test:

- Student.java
- Course.java
- Date.java

Student Class: Constructors

```
// Default constructor
public Student()
{
    // initialise instance variables
    birthDate = new Date();
    courseList = new Course[MAX_COURSES];
}

/** Constructor with roll number */
public Student(String roll)
{
    rollNum = roll;
    birthDate = new Date();
    courseList = new Course[MAX_COURSES];
}
```

Batch midsem marks: Highest

```
public static void main (String arg[])          // code as written in class
{

  int[] rollNum= {6016, 6024, 6078};
  double[] midsemMarks = {61.7,  54 , 74.2 };
  String[] name = {"Abhishek Murthy", "Abhishek Singh", "Ankit Mukund"};

  Student [] batch = new Student[3];
  for (int i=0; i<batch.length; i++)
  {
   batch[i] = new Student();
   batch[i].rollNum = "Y8"+rollNum[i];
   batch[i].setNames(name[i]);
   batch[i].marks = midsemMarks[i];
  }
  System.out.println (Arrays.toString(batch));
  // this prints each record in the array using Student.toString()

  double marksMax = batch[0].marks;
  int index = 0;
  for (int i=1; i<batch.length; i++)
  {
   if (batch[i].marks > marksMax)
    {
          index=i;
          marksMax = batch[i].marks;
    }
  }

  System.out.println ("HIGHEST MARKS:");
  System.out.println (batch[index]);

}
```

Multidimensional Arrays

```
int[][] sqr = new int[5][5];

// this prints out the array
for (int r=0; r < sqr.length; r++) {
   System.out.println(Arrays.toString( sqr[r]));
}
```

Note: Array elements get default values. E.g. numericals get zero.

Can also initialize multidim arrays:

```
int [] [] magicSquare = { {8, 1, 6} , {3, 5, 7}, {4, 9, 2} };
```

Non-uniform rows Arrays

```
int[][] tri;
tri = new int[5][];

for (int r=0; r < tri.length; r++)
{
  tri[r] = new int[r+1];
         // Allocate each row with differing size
}

for (int r=0; r<tri.length; r++)
{
   System.out.println(Arrays.toString(tri[r]));
}
```

Pingala's Triangle

What does this code do?
(nCr is the n choose r function you wrote some weeks back).

```
public static int[][] pingalasTri(int n)
    {
        int[][] tri = new int [n][];
        for (int i = 0; i< n; i++)
        {
            tri[i] = new int[i+1];
            for (int j = 0; j<i+1; j++)
            {
                tri[i][j] = nCr(i,j);
            }
        }
        return tri;
    }
```

See the File: Array2D.java